

Terraform Task

Name: Shaik Khaja Basha

Batch : Batch 11

Date : 21.07.2025

Task : Injecting Variables for diff resources of random

1. inject values from different modes to different resources of random

ans:

1. use different sources of input values (CLI, env vars, tfvars, auto.tfvars, default, etc.)
2. inject those values into **different random resources** like random_pet, random_string, and random_id.

Inject values from the **different input modes**, each targeting to a **different resource**:

❖ Decleration of Variables:

```
variable "pet_prefix" {
  description = "Prefix for random_pet"
  type       = string
  default    = "defaultpet"
}

variable "id_prefix" {
  description = "Prefix for random_id"
  type       = string
}

variable "string_length" {
  description = "Length for random_string"
  type       = number
}

variable "shuffle_input" {
  description = "List for random_shuffle"
  type       = list(string)
}

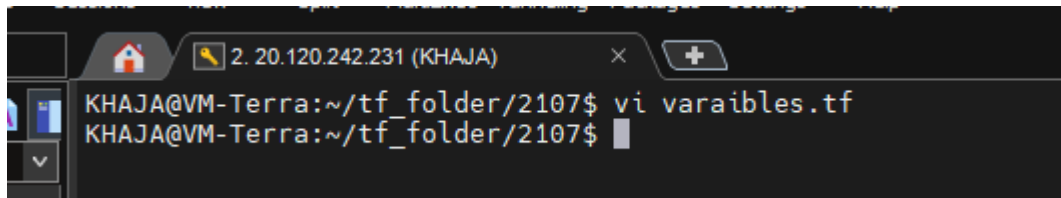
variable "uuid_keepers" {
  description = "Value used to trigger regeneration"
```

```

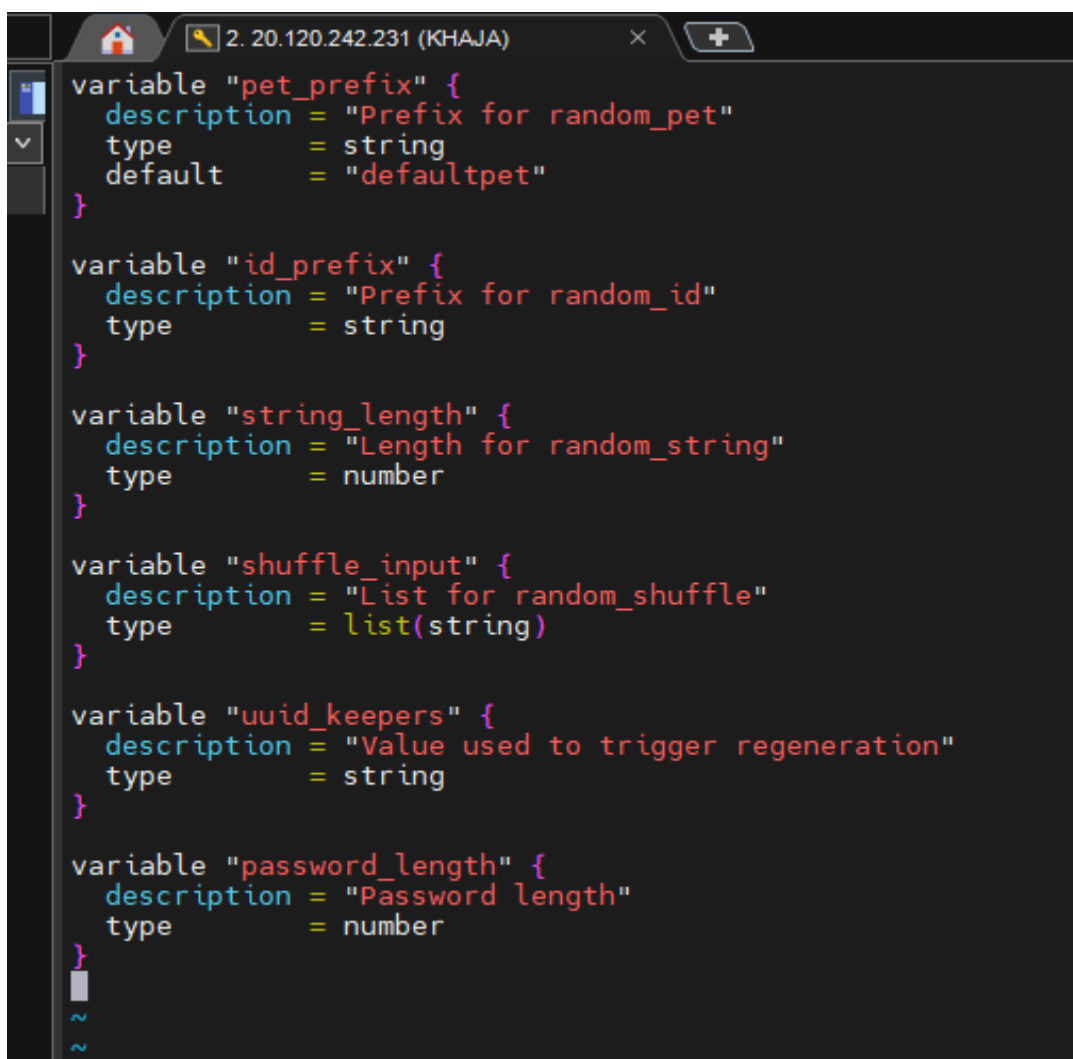
    type    = string
}

variable "password_length" {
    description = "Password length"
    type        = number
}

```



A terminal window with a dark background. The title bar shows a home icon, a magnifying glass, and the IP address 2. 20.120.242.231 (KHAJA). The terminal text shows the user KHAJA@VM-Terra at the directory ~/tf_folder/2107\$ running the command vi variables.tf. The prompt changes to KHAJA@VM-Terra:~/tf_folder/2107\$ after the command is executed.



A code editor window with a dark background. The title bar shows a home icon, a magnifying glass, and the IP address 2. 20.120.242.231 (KHAJA). The editor displays the following Terraform code for variables.tf:

```

variable "pet_prefix" {
    description = "Prefix for random_pet"
    type        = string
    default     = "defaultpet"
}

variable "id_prefix" {
    description = "Prefix for random_id"
    type        = string
}

variable "string_length" {
    description = "Length for random_string"
    type        = number
}

variable "shuffle_input" {
    description = "List for random_shuffle"
    type        = list(string)
}

variable "uuid_keepers" {
    description = "Value used to trigger regeneration"
    type        = string
}

variable "password_length" {
    description = "Password length"
    type        = number
}

```

I will be assigning output variable in the same file variable.tf

```

    }

    output "pet_id" {
      value = random_pet.pet.id
    }

    output "random_string" {
      value = random_string.string.result
    }

    output "shuffled_items" {
      value = random_shuffle.shuffle.result
    }

    output "uuid" {
      value = random_uuid.uuid.result
    }

    output "password" {
      value = random_password.passwd.result
    }
  }

```

Know create Resource Definitions

1. From default

```

resource "random_pet" "pet" {
  prefix = var.pet_prefix
}

```

2. From terraform.tfvars

```

resource "random_id" "id" {
  prefix    = var.id_prefix
  byte_length = 4
}

```

3. From adi.auto.tfvars

```

resource "random_string" "string" {
  length = var.string_length
  special = false
}

```

4. From env variable

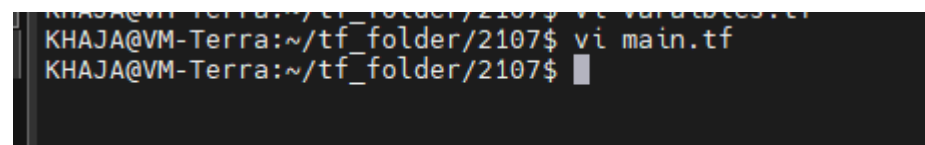
```
resource "random_shuffle" "shuffle" {  
  input      = var.shuffle_input  
  result_count = 2  
}
```

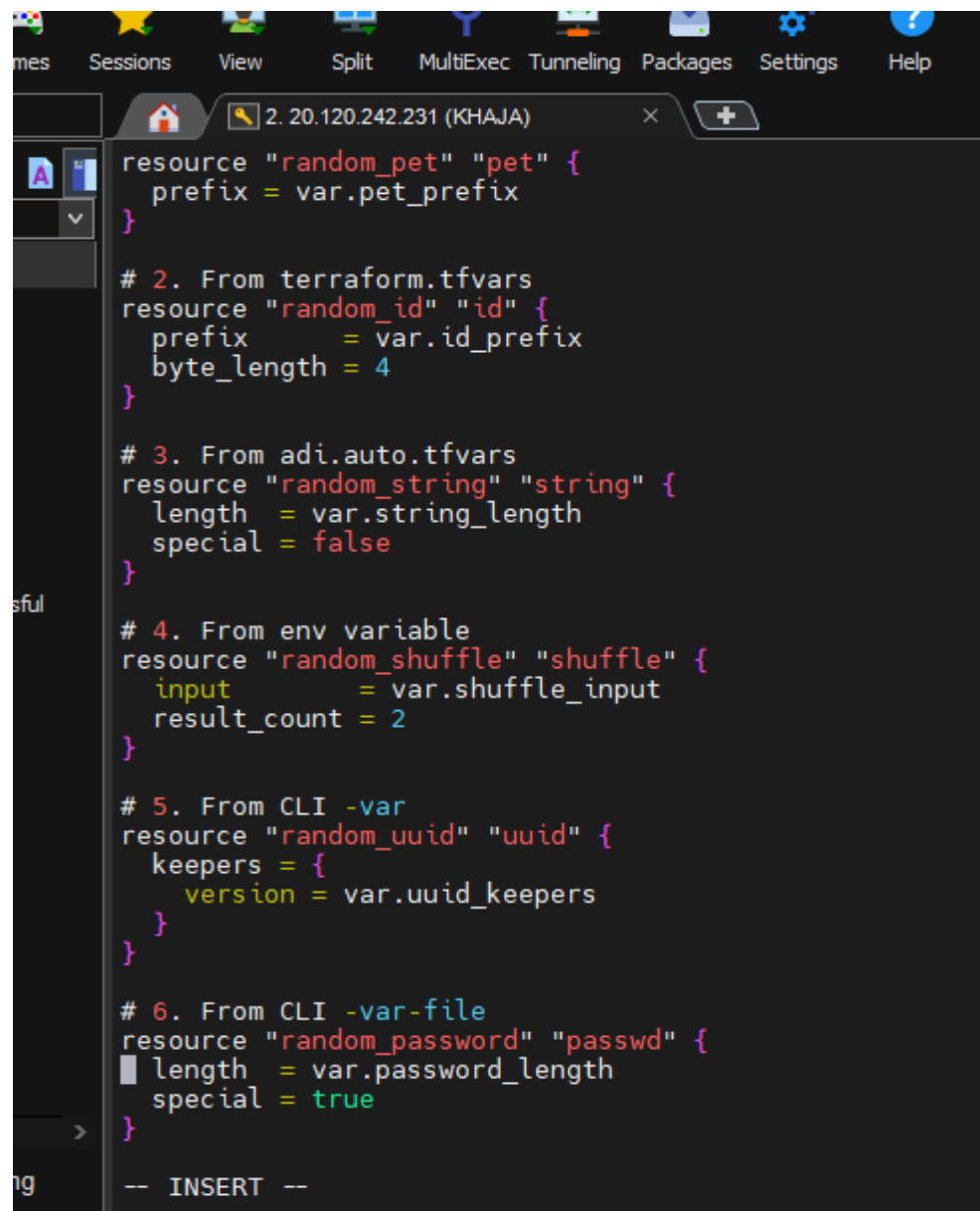
5. From CLI -var

```
resource "random_uuid" "uuid" {  
  keepers = {  
    version = var.uuid_keepers  
  }  
}
```

6. From CLI -var-file

```
resource "random_password" "passwd" {  
  length = var.password_length  
  special = true  
}
```

A terminal window screenshot with a dark background. The prompt is 'KHAJA@VM-Terra:~/tf_folder/2107\$'. The user has entered 'vi main.tf' and the prompt is now 'KHAJA@VM-Terra:~/tf_folder/2107\$' with a cursor. The top of the terminal shows a partial command 'KHAJA@VM-Terra:~/tf_folder/2107\$ vi variables.tf'.



```
resource "random_pet" "pet" {
  prefix = var.pet_prefix
}

# 2. From terraform.tfvars
resource "random_id" "id" {
  prefix      = var.id_prefix
  byte_length = 4
}

# 3. From adi.auto.tfvars
resource "random_string" "string" {
  length    = var.string_length
  special   = false
}

# 4. From env variable
resource "random_shuffle" "shuffle" {
  input       = var.shuffle_input
  result_count = 2
}

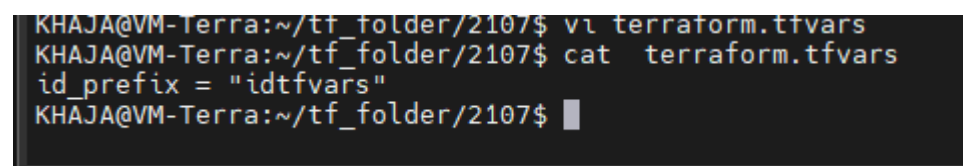
# 5. From CLI -var
resource "random_uuid" "uuid" {
  keepers = {
    version = var.uuid_keepers
  }
}

# 6. From CLI -var-file
resource "random_password" "passwd" {
  length    = var.password_length
  special   = true
}

-- INSERT --
```

❖ terraform.tfvars – Value for random_id

id_prefix = "idtfvars"



```
KHAJA@VM-Terra:~/tf_folder/2107$ vi terraform.tfvars
KHAJA@VM-Terra:~/tf_folder/2107$ cat terraform.tfvars
id_prefix = "idtfvars"
KHAJA@VM-Terra:~/tf_folder/2107$
```



```
id_prefix = "idtfvars"
```

❖ **shaik.auto.tfvars – Value for random_string**

string_length = 10

```
KHAJA@VM-Terra:~/tf_folder/2107$ vi shaik.auto.tfvars
KHAJA@VM-Terra:~/tf_folder/2107$ cat shaik.auto.tfvars
string_length = 10
KHAJA@VM-Terra:~/tf_folder/2107$
```

❖ **Export Env Variable for random_shuffle**

export TF_VAR_shuffle_input='["apple", "banana", "cherry", "dates"]'

```
string_length = 10
KHAJA@VM-Terra:~/tf_folder/2107$ export TF_VAR_shuffle_input='["apple", "banana", "cherry", "dates"]'
KHAJA@VM-Terra:~/tf_folder/2107$ vi prod.properties
KHAJA@VM-Terra:~/tf_folder/2107$ cat prod.properties
```

❖ **Create CLI -var-file for random_password**

prod.properties

password_length = 12

```
KHAJA@VM-Terra:~/tf_folder/2107$ cat prod.properties
password_length = 12
KHAJA@VM-Terra:~/tf_folder/2107$
```

❖ **Run Terraform Apply**

terraform init

terraform apply \

-var="uuid_keepers=fromcli" \

-var-file="prod.properties"

```
password_length = 12
KHAJA@VM-Terra:~/tf_folder/2107$ terraform init
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/random from the dependency lock file
- Using previously-installed hashicorp/random v3.7.2

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
KHAJA@VM-Terra:~/tf_folder/2107$
```

```
See 'snap info <snapname>' for additional versions.
KHAJA@VM-Terra:~/tf_folder/2107$ terraform apply -var="uuid_keepers=cli-uuid-trigger" -var-file="prod.properties"

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following
symbols:
+ create

Terraform will perform the following actions:

# random_id.id will be created
+ resource "random_id" "id" {
  + b64_std      = (known after apply)
  + b64_url      = (known after apply)
  + byte_length = 4
  + dec          = (known after apply)
  + hex          = (known after apply)
  + id           = (known after apply)
  + prefix       = "idtfvars"
}

# random_password.passwd will be created
+ resource "random_password" "passwd" {
  + bcrypt_hash = (sensitive value)
  + id          = (known after apply)
  + length      = 12
  + lower       = true
  + min_lower   = 0
  + min_numeric = 0
  + min_special = 0
  + min_upper   = 0
  + number      = true
  + numeric     = true
  + result      = (sensitive value)
}
```

```
2. 20.120.242.231 (KHAJA) x +

# random_pet.pet will be created
+ resource "random_pet" "pet" {
  + id          = (known after apply)
  + length      = 2
  + prefix       = "defaultpet"
  + separator    = "-"
}

# random_shuffle.shuffle will be created
+ resource "random_shuffle" "shuffle" {
  + id          = (known after apply)
  + input       = [
    + "apple",
    + "banana",
    + "cherry",
  ]
  + result      = (known after apply)
  + result_count = 2
}

# random_string.string will be created
+ resource "random_string" "string" {
  + id          = (known after apply)
  + length      = 10
  + lower       = true
  + min_lower   = 0
  + min_numeric = 0
  + min_special = 0
  + min_upper   = 0
  + number      = true
  + numeric     = true
  + result      = (known after apply)
  + special     = false
  + upper       = true
}

VM-Terra 0% 0.39 GB / 0.83 GB 0.01 Mb/s 0.00 Mb/s 56 min KHAJA
```

```
2. 20.120.242.231 (KHAJA) x +
# random_uuid.uuid will be created
+ resource "random_uuid" "uuid" {
  + id      = (known after apply)
  + keepers = {
    + "version" = "cli-uuid-trigger"
  }
  + result = (known after apply)
}

Plan: 6 to add, 0 to change, 0 to destroy.

Changes to Outputs:
+ password      = (sensitive value)
+ pet_id        = (known after apply)
+ random_string = (known after apply)
+ shuffled_items = (known after apply)
+ uuid          = (known after apply)

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

random_id.id: Creating...
random_id.id: Creation complete after 0s [id=i0wzoQ]
random_shuffle.shuffle: Creating...
random_password.passwd: Creating...
random_string.string: Creating...
random_pet.pet: Creating...
random_uuid.uuid: Creating...
random_shuffle.shuffle: Creation complete after 0s [id=-]
random_string.string: Creation complete after 0s [id=WHiHTUCycv]
random_pet.pet: Creation complete after 0s [id=defaultpet-firm-dodo]
random_uuid.uuid: Creation complete after 0s [id=3ce7c1e9-6a45-a901-c3f6-bed54c617a15]
```

```
random_pet.pet: Creation complete after 0s [id=defaultpet-firm-dodo]
random_uuid.uuid: Creation complete after 0s [id=3ce7c1e9-6a45-a901-c3f6-bed54c617a15]
random_password.passwd: Creation complete after 0s [id=none]

Apply complete! Resources: 6 added, 0 changed, 0 destroyed.

Outputs:

password = <sensitive>
pet_id = "defaultpet-firm-dodo"
random_string = "WHiHTUCycv"
shuffled_items = tolist([
  "apple",
  "cherry",
])
uuid = "3ce7c1e9-6a45-a901-c3f6-bed54c617a15"
KHAJA@VM-Terra:~/tf_folder/2107$
```